$1.50

February, 1985
Vol.5 No.2

M.A.C.E. JOURNAL

"Devoted Exclusively To The Atari Computer User"

IN THIS ISSUE...

E*X*T*E*N*D*E*D BASIC

ANTIC'S C.E.S. ATARI PREVIEW

BANK-SWITCHING CARTRIDGES

XMODEM PROTOCOL

REVIEWS

MODEMS AND AMODEM

LOGO DISPLAY LIST INTERRUPTS

...AND MORE!

Published by the Michigan Atari Computer Enthusiasts

FIRESIDE CHAT

Well, here I am feeling happy about my Atari. I read the report on CES printed elsewhere in this Journal, and was really impressed. I can't wait to get my hands on a 3 1/2" 500k disk drive! And for only $100!!!

One bad thing about Atari is that they don't advertise. The Vice President of Atari, James Copeland, was on WXYT Radio's "Tech Talk" a couple weeks ago and I got in as the first, and only, caller to question him. Advertising is what I asked him about. It's sad, but Atari feels it's not their place to advertise. They feel it's their job to manufacture and the retailers' (Sears, Toys-R-Us, etc.) to advertise. The only way to push our product is to do it ourselves. Maybe, just maybe, we should start a big letter-writing campaign and let the powers that be know how we feel. It makes me mad, as a proud Atari owner, to watch TV and see 3 or 4 Commodore ads, a couple of Apple ads, and nothing about Atari. What good is it to make a super machine and not tell anyone?

Anyhow, enough rambling from me. There is enough good stuff to read in here this month without my going on and on. Remember, write to Atari!!

Kirk

The MACE Journal is published monthly by the Michigan Atari Computer Enthusiasts. Unless otherwise noted, material published in the Journal is in the public domain and may be reproduced for private or user group use providing proper credit is given.

Submissions to the Journal can be mailed to the PO Box, uploaded to the MACE BBSs or any officer's BBS, or uploaded directly to the editor at 646-4455. Where possible, submissions should include a disk or tape file in AtariWriter or similar format and a working copy of the program. Specify format for screen dumps (AtariArtist, Koalapad, etc.). Authors whose submissions are published will receive a certificate good for a free disk or tape from the MACE library.

ATARI SERVICE CENTERS

Antic magazine lists the following Atari service centers in Michigan:

The Family Computer Center
3895 W. 12 Mile Rd.
Berkley
(313) 543-0520

ABL Electronic Service, Inc.
32 E. 14 Mile Rd.
Madison Heights
(313) 588-6663

Solid State Service
548 Shattuck Rd.
Saginaw
(517) 752-0606

Future Directions
1520 N. Van Dyke
Bad Axe
(517) 269-7211

Chase Transistor Service
521 Leonard St. NW
Grand Rapids
(616) 454-9000

Remember that with Atari's new service policy, equipment under warranty must be returned to the place of purchase for replacement. After the 90-day warranty period, repairs are at the expense of the owner.

IS YOUR TIME UP?

Please take a moment to look at the mailing label on the back cover of this Journal. Are your name and address correct? And what about that date in the upper right-hand corner? That's the expiration date of your MACE membership. If it's coming up soon, plan to renew at the next MACE meeting, or fill out the membership form on the inside back cover and send it in.

EXTENDED BASIC

INTRODUCTION

Extended BASIC is a programmer's aid package which adds twelve useful commands to BASIC. The commands are direct mode commands and function as if they were included in the BASIC cartridge. There are DOS-related, editing, reference and control commands.

SYSTEM REQUIREMENTS

Atari 400/800
(Translator disk required for XL machines)
32K RAM
Atari 810 disk drive or equivalent
Atari BASIC language cartridge
System printer (Optional)

LOADING EXTENDED BASIC

1. Turn the computer off.
2. Insert the BASIC cartridge.
3. Turn on disk drive #1.
4. When the busy light goes out, insert the Extended BASIC diskette.
5. Turn on the computer and TV. The program will load into memory and start automatically.
6. After the system boots the top line on the screen should say:

   Extended BASIC 1.0

You may now edit and RUN BASIC programs normally.

USING EXTENDED BASIC

Extended BASIC commands must be entered in direct mode and there can be only one command per line. All direct mode input to BASIC is converted to upper-case normal video. The only text that is not converted is the text within double quotes.

COMMANDS

The following commands comprise the Extended BASIC program:

EDITING: RENUMBER, DELETE, REPLACE, MARGIN
DOS UTILITIES: DIRECTORY, LOCK, UNLOCK, ERASE, RENAME
REFERENCE: QREF, XREF, SIZE
CONTROL: @D, @E, @X

EDITING COMMANDS

RENUMBER (REN.): Takes the program text and renumbers it according to the parameters entered. Extended BASIC will renumber the program even if the program contains variable branching and unreferenced line numbers. Variable branching and unreferenced line numbers are displayed during the renumbering process.

FORMAT: RENUMBER [STRT] [,NEW] [,INC] [,OUTPUT]
Parameter explanation:

STRT - Starting line number (default is the first line of the program)

NEW - New starting line number (default is the starting line number)

INC - Increment between line numbers (default is 10)

OUTPUT - Device to which the error flags will be printed. Valid output devices are E:, S: and P: (default is E:)

Examples:

RENUMBER 10,,20,"P:"

The program will be renumbered starting at 10 with increments of 20; the error flags will go to the system printer.

REN. ,,5

If the first program line was 100 then the program will be renumbered starting at 100 with increments of 5; the error flags will go to the screen.

REN. ,1000,,"E:"

The program will now start at 1000 with increments of 10; the error flags will go to the screen.

RENUMBER ERROR FLAGS

The error flags are printed whenever a variable branch or an unreferenced line number is encountered. The types of error flags are as follows:

V/1000 - The V indicates a variable branch and 1000 is the line number where it occurs.

#/40000/1000 - The # indicates an unreferenced line number, the 40000 is the unknown line number and the 1000 is the line number where it occurs.

DELETE (DEL.): Deletes all line numbers within the two parameters.

FORMAT: DELETE STRT [,END]
Parameter explanation:
  STRT - Starting line number
  END - Ending line number (default is the starting line number)
Examples:
  DELETE 10,100
Lines 10 through 100 are deleted.

  DEL. 20
Line 20 is deleted.

REPLACE (REPL.): Replaces a variable name with a new variable name.

FORMAT: REPLACE "OLD-VAR", "NEW-VAR"

Parameter explanation:
  OLD-VAR - Old variable name. This must be a valid variable name.
  NEW-VAR - New variable name. This must be a valid variable name and unique.
If OLD-VAR is a string then NEW-VAR must be a string. The same applies to array names. The format for variable names is described in the BASIC Reference Manual.
Examples:
  REPLACE "A$","B$"
All occurences of A$ are replacesd with B$ and A$ is removed from the variable name table.

  REPL. "ARRAY(","DARRY("
All occurences of ARRAY(are replaced with DARRAY(and ARRAY(is removed from the variable name table.

  REPL. "COUNTER","LCOUNT"
All occurences of counter are replaced with LCOUNT and COUNTER is removed from the variable name table.

REPLACE ERROR FLAGS
  22 - Old variable name not found.
  23 - New variable name already exists.
  24 - Invalid variable name.

MARGIN (MAR.): Automatically sets the left and right margins.

FORMAT: MARGIN [LEFT][,RIGHT]
Parameter explanation:
  LEFT - Left margin; this must be greater than or equal to 0 and less than 40 (default is 2)
  RIGHT - Right margin; this must be greater than or equal to 0 and less than 40 (default is 39)

Examples:
  MAR. 0,39
The margins are set to the extreme left and right edges of the screen.
  MAR.
The margins are set as they are after system reset.

DOS UTILITIES

Five DOS commands are included in the Extended BASIC program. They will allow the programmer limited file maintenance. Note: The DOS command has been disabled to give the programmer the most free memory, and because DUP will overwrite Extended BASIC's memory. If you need to go to DOS the following commands will get you there:

  @X
  DOS

DIRECTORY (DI.): Prints the directory of a disk to the output device.

FORMAT: DIRECTORY [DRIVE#][,OUTPUT]
Parameter explanation:
  DRIVE# - Disk drive number for the desired directory (default is drive #1)
  OUTPUT - Device to which the printed output is to go. Valid devices are E:, S: and P: (default E:)

Examples:
  DIR.
The directory of drive #1 is displayed on the screen.

  DIR. ,"P:"
The directory of drive #1 is printed on the system printer.

  DIRECTORY 2
The directory of drive #2 is displayed on the screen.

LOCK, UNLOCK (UNL.) AND ERASE (ERA.): These commands lock, unlock and erase (delete) files on a disk.

FORMAT: LOCK "D[N]:FILESPEC[. EXT]"
This command will lock the selected file, or files that meet wildcard specifications.

FORMAT: UNLOCK "D[N]:FILESPEC[.EXT]"
This command will unlock the selected file, or files that meet wildcard specifications.

FORMAT: ERASE "D[N]:FILESPEC[.EXT]"
This command will erase the selected file, or files that meet wildcard specifications.

Parameter explanation:
N - Drive number (default is drive #1)
FILESPEC - Filename & optional extension

RENAME (RENA.): Renames a file with a new file name.

FORMAT: RENAME "D[N]:OLDNAME.EXT, NEWNAME.EXT"

Parameter explanation:
N - Drive number (default is drive #1)
OLDNAME - The old filename
NEWNAME - The new filename

REFERENCE COMMANDS

QREF (QR.): QUICK REFERENCE gives a dump of the variable name table. When the screen fills up, press any key to continue to the next screen.

FORMAT: QREF [OUTPUT]

Parameter explanation:
OUTPUT - Device to which the printed output is to go. Valid devices are E:, S: and P: (default device is E:

XREF (XR.): CROSS REFERENCE gives a cross reference of all variables used in the program. It will print the variable name and all occurences of it.

FORMAT: XREF [OUTPUT]

Parameter explanation:
OUTPUT - Device to which the printed output is to go. Valid devices are E:, S: and P: (default device is E:)

Example:
XREF "S:"

A$
2/100 1/110 10/1000

A$ is the variable name and in the second line the 2/100 indicates 2 occurences in line number 100. The 1/110 indicates 1 occurence in line 110, and so on.

SIZE (SI.): This command requires no parameters. It will return the number of variables in the program, the program length and the amount of free space remaining.

FORMAT: SIZE

OUTPUT FORMAT:
Number of variables / Program size / Free memory remaining

Example:
SIZE
12/500/28500

This indicates that there are 12 variables in the variable name table. The program length is 500 bytes and there are 28500 bytes of memory remaining.

CONTROL COMMANDS

@D: Temporarily DISABLEs Extended BASIC. It should be used before running any program that contains any input statements.

FORMAT: @D

@E: ENABLEs Extended BASIC. It is the power-up mode.

FORMAT: @E

@X: TERMINATEs Extended BASIC in case it is not required or if more memory is needed. A "NEW" command should be executed after @X to reclaim memory.

FORMAT: @X

REFERENCE CARD

```
  COMMAND    ABBREV.  PARAMETERS

EDITING

  RENUMBER   REN.     [STRT][,NEW][,INC][,OUTPUT]
  DELETE     DEL.     STRT[,END]
  REPLACE    REPL.    "OLD-VAR","NEW-VAR"
  MARGIN     MAR.     [LEFT][,RIGHT]

DOS UTILITIES

  DIRECTORY  DI.      [DRIVE#][,OUTPUT]
  LOCK       LOCK     "D[N]:FILESPEC[.EXT]"
  UNLOCK     UNL.     "D[N]:FILESPEC[.EXT]"
  ERASE      ERA.     "D[N]:FILESPEC[.EXT]"
  RENAME     RENA.    "D[N]:OLDNAME.EXT,NEWNAME.EXT"

REFERENCE COMMANDS

  QREF       QR.      [OUTPUT]
  XREF       XR.      [OUTPUT]
  SIZE       SI.

CONTROL COMMANDS

  @D     TEMPORARILY DISABLE EXTENDED BASIC
  @E     RENABLE EXTENDED BASIC
  @X     TERMINATES EXTENDED BASIC
```

CREATING THE FILE

Type in the BASIC loader program exactly as printed and SAVE a copy. Put a formatted disk with DOS files in Drive #1, then RUN the program. It will create a file called AUTORUN.SYS on the disk in Drive #1. Follow the directions above to use Extended BASIC.

BAD JOKE DEPARTMENT

Here's a groaner for all the mathematicians in the crowd:

Why do programmers get Christmas and Halloween confused?

--Because 25 DEC = 31 OCT

```
610 DIM A$(100),HLD$(54),PT$(51):RESTO
RE :N=50
620 FOR I=1 TO 54:READ X:HLD$(I)=CHR$(
X):NEXT I
625 FOR I=1 TO 51:READ X:PT$(I)=CHR$(X
):NEXT I
630 GRAPHICS 0:POKE 710,0:POKE 709,14:
LIST 640,650
635 GOSUB 750
640 REM .         Extended Basic
644 REM .         by Jim Nangano
650 OPEN #1,8,0,"D1:AUTORUN.SYS"
651 CD=80
655 POKE 82,0:POKE 83,39:? :? "Records
 left to write: ";:POKE 755,0
660 TRAP 900:READ A$:TRAP 40000:? CD;"
  ":? CHR$(28);CHR$(127);CHR$(127);CHR
$(127);:CD=CD-1
670 Y=USR(ADR(HLD$))
680 IF LEN(A$)<50 THEN N=INT((LEN(A$)+
0.5)/2)
690 X=USR(ADR(PT$),ADR(A$),N)
700 GOTO 660
750 ADDR=ADR(A$):HI=INT(ADDR/256):LO=A
DDR-(HI*256)
760 HLD$(7)=CHR$(LO):HLD$(25)=CHR$(LO)
:HLD$(41)=CHR$(LO)
770 HLD$(8)=CHR$(HI):HLD$(26)=CHR$(HI)
:HLD$(42)=CHR$(HI)
780 ADDR=ADR(HLD$)+52:HI=INT(ADDR/256)
:LO=ADDR-(HI*256)
790 HLD$(21)=CHR$(LO):HLD$(38)=CHR$(LO
)
800 HLD$(22)=CHR$(HI):HLD$(39)=CHR$(HI
)
810 RETURN
900 CLOSE #1:? :? "Done.":? :? :? :?
909 POKE 755,2:END
910 DATA 104,162,0,160,0,189,2,1,201,6
4,48,3
920 DATA 24,105,9,10,10,10,10,141,4,3,
232,189
930 DATA 2,1,201,64,48,3,24,105,9,41,1
5,24
940 DATA 109,4,3,153,2,1,200,192,50,20
8,1,96
950 DATA 232,184,80,209,0,0
960 DATA 104,104,133,207,104,133,206,1
04,104,133,204,169,0,133,205,162,16,16
9,11,157,66,3,169,0,157
970 DATA 72,3,157,73,3,164,205,177,206
,32,86,228,132,195,16,1,96,230,205,165
,205,197,204,208,223,96
1010 DATA FFFF0F1F671FA50C8D531FA50D8D
```

```
541FA207A97D9D1A03E8A91F9D1A03A9728DE7
02A92E8DE802A919850CA91F850DA508F0
1020 DATA 0420501F60206D2C20552C20E22C
20612C207D2C606C531F0000457874656E6465
64204261736963203 12E309B7D1F7820FB
1030 DATA F333F68B1FA3F633F63CF64CE4F3
8DA12BA91F48A99A48ADA12B4C3EF608489848
8A4BADFB02C99BD025A00912420D41FEA
1040 DATA EAEA20222020732 0B013BDC22085
D1204D2190062043224CCD1F20822268AA68A8
682860A000B1F3C940D02BC8C8B1F3C99B
1050 DATA D02388B1F3C944F00AC945F019C9
58F022D012A94C8DAF1FA9CD8DB01FA91F8DB1
1F20482B60A9EA8DAF1F8DB01F8DB11FD0
1060 DATA EFAD531F850CAD541F850D4C74E4
A000A97F8D3520A9208D4820B1F3C99BF01D29
7FC922D003205220C9619008C97BF004B0
1070 DATA 02492091F3C830034C2E206048AD
3520C97FF00DA97F8D3520A9208D48204C7120
A9FF8D3520A9008D48206860A200A000BD
1080 DATA C2792074212030 2FC92EF009D1F3
D027C8E84C7720E8B1F3C92EF01088BDC22030
0AC8D1F3D010C8E84C8A20BDC22008E828
1090 DATA 10F8C81860BDC22008E82810F8E8
E8E8BDC220C91BF0034C752038605 2454E2E55
4D424552800F8A2244454C2E4554458003
1100 DATA 222644492E524543544F52598009
E82652454E412E4D458020A6274552412E5345
8021A6274C4F434B2E8023A627554E4C2E
1110 DATA 4F434B8024A62751522E45468008
E3275245502E4C4143458040 1A2858522E4546
8008B22953492E5A458000B62A4D41522E
1120 DATA 47494E8003232B444F2E53800047
281B1B8A4820022D20122DA5D1F01F2940D01B
A5D12920D015A5D12907D026A5D12908F0
1130 DATA 09206D2C20552C752170224CF321
A90085F268AAE81860A90085F2B1F3098091F3
68AA38608DA02BA9018D4222A2008E9F2B
1140 DATA B1F3C920F023C92CF00FC99BF0BF
C93090D3C93A901D4C81214EA02BF0AF0E4222
AE9F2BE8E88E9F2BB1F3C99BF09EC84C9C
1150 DATA 2184F22000D8A5F24820D2D9AE9F
2BA5D49D912BA5D59D922B68A8A5D14D422285
D14CB521B1F3C92CF00FC99BF021C920F0
1160 DATA 07C922F0074C8121C84CF321C8B1
F3C950F013C945F007C953F0034C8121A9008D
41224C7821A5D1498085D1204A2C20C22C
1170 DATA 100A68686898202 62D4CCD1F4C78
210000A98085CBA90585CCA98985CDA90585CE
A96E85CFA90085D020042EA008B93D2D91
1180 DATA F38810F8206D2C20552CA200A971
226C23099D42032056E4207D2C20612C20482B
60BDC32048BDC2204860A9008DA22BAD92
1190 DATA 2B3045AD942B3040AD962B303BA5
D12901F01020092CA000B1CB8D912BC8B1CB8D
922BA5D12902F00CAD912B8D932BAD922B
1200 DATA 8D942BA5D12904F012A90A8D952B
A9008D962B4CDF22A9114C262DAD952B0D962B
F0E6A99B8D551F20E22C20092C20B22B90
1210 DATA 034CA22A20B82520B22B90034C21
23A000B1CB91CDC8B1CB91CDC8A90091CDC891
CD20AA2520A32B4CFD22A000A9FF91CDC8
1220 DATA 91CD20B825A001B1CD8D032E88B1
CD8D022EAD922B8D012EAD912B8D002E20DF2D
90034C6E23A000B1CDA00291CDA001B1CD
1230 DATA A00391CDA9018DA22B20AA2520C4
2590C24C226D23682425ADA22BF03138A5CDE9
0485CDA5CEE90085CEA003B1CD8D032E88
1240 DATA B1CD8D022EAD942B8D012EAD932B
8D002E20DF2D90034C2A2520AA25A002AD932B
91CDAD942BC891CD20AA2518AD932B6D95
1250 DATA 2B8D932BAD942B6D962B8D942B10
034C252520C42590D320092C20B82518A5CD69
0285CFA5CE690085D0A000B1CF91CBC8B1
1260 DATA CF91CBC8B1CB8D9E2BC818A5CF69
0485CFA5D0690085D0B1CB8D9D2BC8B1CBC90A
9004C90E9023C91EF049C923F01BC907D0
1270 DATA 034CF824AD9D2BCD9E2BF004A84C
032420A32B90B14C2225C8B1CBC90ED01DC898
48186906A8B1CBC916F008C914F004684C
1280 DATA 5A2468A8202F254C222420E5244C
2224C8B1CBC917F01BC91869246425F017C90E
F007C90FF0094C602420122C8890E42019
1290 DATA 2C8890DEC8B1CBC90ED02C984818
6907A8B1CBC912F00CC916F008C914F004684C
B52468A8C8202F2598186906A8B1CBC912
1300 DATA F0D04C2224B1CBC912F00EC914F0
04C916D00C20E5244C222420E5244C8224C90E
F008C90FF009C84CB52420122C90D52019
1310 DATA 2C90D09848A000B1CB85D4C8B1CB
85D568A820F42560AC9D2B88B1CBC914F007C9
16F0034C1F259838E908A8B1CBC91BD00B
1320 DATA C8B1CBC90ED004C8202F254C2224
4C922AA9074C262DA9114C262D9848 1865CBAA
A5CC6900A82089DD20D2D9A5D58D032EA5
1330 DATA D48D022EAD922B8D012EAD912B8D
002E20DF2D904920C425B015A000B1CDC56525
6026D4D007C8B1CDC5D5F01920AA254C5B
1340 DATA 25A000B1CB8D9B2BC8B1CB8D9C2B
20D5254CA425A002B1CD85D4C8B1CD85D520AA
D968481865CBAAA5CC6900A820A7DD20B8
1350 DATA 2568A86018A5CD690485CDA5CE69
0085CE60A58C85CD18A58D690185CE60A000A9
FFD1CDD007C8D1CDD0023860 1860AE2226
1360 DATA A9239D551FE89848A90285D120D5
2BAD9B2B85D4AD9C2B85D54C0326AE2226A956
9D551FE89848A90285D120D52B18AD2226
1370 DATA 69128D2226C924900CAAA99B9D55
```

```
1F20E22C20022D68A86000A5D12901F00160A5
D12902F00CAD912B8D932BAD922B8D942B
1380 DATA 20092C20B22B90034CAB2620AF26
90034C572620A32B90F34CAB26A5CB85CDA5CC
85CE20CC6126SC27269005F00B4C742620
1390 DATA A32B90F14C742620A32B38A590E5
CB85CFA591E5CC85D038A5CBE5CD8D972BA5CC
E5CE8D982BA200A00438B58AED972B958A
1400 DATA B58BED982B958BE8E888D0EC2004
2E20482B60A001B1CB8D032E88B1CB8D022EAD
922B8D012EAD912B8D002E20DF2D60A001
1410 DATA B1CB8D032E88B1CB8D022EAD942B
8D012EAD932B8D002E20DF2D60209527A5D129
01D014AD912BC9019004C9059005A9A04C
1420 DATA 262D6930D002A9318DA127A220A9
A09D4403A9279D4503A9039D4203A9069D4A03
2056E43048A0008C22262D022DA0002072
1430 DATA 27301CAC2226D00DA011A9209955
1FC88C2226D0E9A9008D222620E22C10DAA99B
AC2226F00699551F20E25D2758282C204A
1440 DATA 2C209527207D2C20612C20482B60
984C262DA2209B1869559D4403A91F69009D45
03A9159D4803A9009D4903A9059D420320
1450 DATA 56E460A220A90C9D42032056E460
44313A2A2E2A9B206D2C204A2CA922A00120B5
2DC8981865F39D4403A5F469009D4503A9
1460 DATA 22A00220B52DA99B99F300A5D19D
42032056E43007207D2C20482B6098C262D20
522D20CC2BA0148C9F2B20222C9002B010
1470 DATA 20E22C3018CE9F2B1003205D2D4C
EF27204A2C20612C207D2C20482B6098C980F0
EE4C262DA2008AA8C8A92220B52D90034C
1480 DATA 5229C8989D912BE8E004D0E9CE92
2BCE942BAC932BB1F3C9419004C95B9005A918
4C262DC8CC942BF020B1F3C95BB0EFC959
1490 DATA 28542941B0F0C93AB0E7C931B0E8
C928F004C924D0DBC8CC942BD0D5AC922B88B1
F3C924F004C928D00BAC942B88D1F3F01C
1500 DATA 4C4728AC942B88B1F3C924F004C9
28D00BAC922B88D1F3F0034C472BAC942B88B1
F3098091F3AC922B88B1F3098091F3AD91
1510 DATA 2B20612990034C5729AD932B2061
29B0034C5C2938AD922BED912B8D952B186D97
2B85CBAD9B2B690085CC38AD942BED932B
1520 DATA 8D962B186D972B85CDAD982B6900
85CE38A590E5CB85CFA591E5CC85D020042EA2
0C38B584ED952B9584B585E90095851885
1530 DATA 846D962B9584B585690095B5CACA
10E0AD9B2B85CBAD9C2B85CCAD972B85CDAD98
2B85CEAD962B85CFA90085D020042E204B
1540 DATA 2B60A9114C5529502A262DA9164C
262DA9174C262D1865F385CF8D9B2BA5F46900
85D08D9C2B20CC2B204E2BB0034CB129B1
1550 DATA CDD1CFD007B1CF301EC810F3B1CD
3003C810F9C8981865CD85CDA5CE690085CE20
4E2B900EB0D8A5CD8D972BA5CE8D982B18
1560 DATA 60386020522D20CC2BA97F8D972B
20222C90034C922AEE972B20E22C10034CAF2A
20022D20092C20B22B90034CBE29A9008D
1570 DATA 9828A000B1CB85CFC8B1CB85D0C8
B1CB8D992BC8B1CB8D9A2BC8B1CBF0363029C8
B1CB3024C90EF016C90FF017C8CC9A2BD0
1580 DATA EEAD9A2BCD992BF019A84CF62920
122C90DD20192C90D8CD972BD0DFEE982B4C0F
2AAD982BF03DAE2226AD982B85D4A90085
1590 DATA D5A90085D120D52BA90285512A4C
2BD1A5CF85D4A5D085D520D52B18AD22266908
8D2226C920900EAAA99B9D551F20E22C30
1600 DATA 3A20022D20A32BB0034CDF29AD22
26D0E5A99B8D551F20E22C30204CBE29AD2226
F00BAAA99B9D551FA20020E22C204A2C20
1610 DATA 7D2C20612C20182B6098C980F0EE
4C262D206D2C20552CA9008D412285D485D520
CC2B20222CB008E6D4D0F7E6D5D0F32002
1620 DATA 2DA20086D120D52B38A58AE58885
D4A58BE58985D5A90285D120D52B38AD3002E5
8A85D4AD3102E58B85D5A90285D120D52B
1630 DATA A99B9D551F20E22C207D2C20612C
A96C85CDA92B85CE208D2C60A5D12901F005A9
028D912BA5D12902F005A9398D932BAD91
1640 DATA 2B8552AD932B855320482B60A92E
8D80054D2B6F2B60A000B1CD297FC9419012C9
5BB00EA5CDC584D006A5CEC585F0023860
1650 DATA 1860434C529B912B8C2C00000000
00000000000000000000000000000A00218B1CB
65CB85CBA5CC690085CCA5CC8D032EA5CB
1660 DATA 8D022EA58B8D012EA58A8D002E20
DF2D601860A58285CDA58385CE60A5D12902F0
06A92F9D551FE898488A48A5D12901D003
1670 DATA 20AAD920E6D868AAA000B1F33007
9D551FC8E8D0F5297F9D551FE868A860A58885
CBA58985CC6018986907A81860C8981871
1680 DATA CBA8C81860204E2B9021B1CD0829
7F99551FC828300210F2981865CD85CDA5CE69
0085CEA99B99551F18603860A210A90C9D
1690 DATA 42032056E460A20FBD40039D812B
CA10F760A20FBD40039D812BCA10F760A20FB5
209D702BCA10F8A5178D802B60A20FBD70
1700 DATA 2B9520CA10F8AD802B851760BD2C
882D201D2DA000B1CD0891F3C82810F7888CA1
2B20A32C6018986D44038524AD45036900
1710 DATA 852538AD4803EDA12B8528AD4903
E900852960A2108E4122A9DF9D4403A92C9D45
03A9089D4A03A9039D42032056E460503A
1720 DATA 9BAE4122A9099D4203A9789D4803
A9009D4903A9559D4403A91F9D45032056E460
A228A9209D551FCA10FAA9008D222660A2
1730 DATA 11A9009D912BCA10FA60A90585F4
```

```
A98085F36085CB207D2C20612CA93B85CDA92D
85CE208D2C38603F224552524F522D2020
1740 DATA 20223B5045454B28323033299BAE
4122D005A97D20A4F660AE4122D042A99B8D55
1FA9099D42032056E4A9A59D4403A92D9D
1750 DATA 4503A9789D4803A9009D4903A909
8D420320892D742E56E4A9FF8DFC02ADFC02C9
FFF0F920522DA9148D9F2BA9FF8DFC0260
1760 DATA 3C505245535320414E59204B4559
3E9B8DDE2D8B84CDA000D1F3F00FA99BD1F3F0
11CB300EADDE2D4CBD2DC6CD3008C84CBD
1770 DATA 2D384CDD2D186000AD022ECD002E
F00BAD032EED012E0901700A60AD032EED012E
70016049800901600000000A5CD38E5CB
1780 DATA AAA5CEE5CCA88AC5CF98E5D0B006
20422E4C202E20212E60A000A6D0F00EB1CB91
CDC8D0F9E6CCE6CECAD0F2A6CFF008B1CB
1790 DATA 91CDC8CAD0F860A5D01865CC85CC
A5D01865CE85CEA4CFF00988B1CB91CDC000D0
F7A6D0F010C6CCC6CE88B1CB91CDC000D0
1800 DATA F7CAD0F060000F1FE002E1020F1F
```

THREE ASSEMBLERS

by Todd A. Meitzner

In this article I will discuss the three most widely used Atari assemblers: Atari's cartridge assembler, OSS's macro assembler "MAC65" (cartridge), and Atari's macro assembler "AMAC".

The first and simplest is Atari's cartridge assembler. While it has the lowest price, it also has the fewest features. (It does however have a debugger, which can help once you have assembled your program.) This assembler is probably a good choice for someone new to assemblers as it is widely used and known. It also has an advantage in that the editor and assembler are together. This means that you can edit a program, assemble it, and if it has errors you can edit again instantly. One of the disadvantages of this assembler is that the source program must be in memory or in one file on disk or cassette. This limits the size of a program you can assemble. It also has a few minor bugs. The manual is acceptable but could be better. There is an addendum available for it.

The OSS "MAC65" is also now available in cartridge form. It used to be available on disk only but was enhanced and put on a cartridge. It is compatible with the Atari cartridge assembler but has a number of additional capabilities, including the ability to assemble a program from more than one file and to define macros. A macro is a user-created instruction in assembler which substitutes for a number of instructions in the unassembled program. MAC65 includes the editor, assembler, and a debugger (it differs from the Atari cartridge assembler debugger). As far as I know no real bugs have been found in it yet. The manual is probably the best of the three as it seems to describe all the features in detail, including the macro features.

The last assembler is the Atari macro assembler "AMAC". The main disadvantage is that the editor and assembler are two separate programs. Each program in itself is very powerful but they must be used separately. This means you must first load the editor, edit the assembler program, save it to disk, load the assembler, assemble the program to disk, and if there are any errors start all over. Another slight disadvantage is that the program is copy-protected; this however can be disabled easily enough. One of the major faults of this assembler is that it is not compatible with the catridge assembler, but if you are willing to accept the incompatibility and the separation of the editor and assembler it is not too bad. In terms of raw power this assembler is probably the most powerful of the three. It too is a macro assembler but is more powerful than MAC65 in this respect. As to bugs in it, the only major one deals with its output to the printer. If you print out to the printer normally it's okay, but if you use an EJECT command or any command which does an eject it will eject out paper and will start the next page in the middle (it supposedly works fine on the Atari 825). It also has a very limited manual for the capabilities that are in it; for the most part you will have to learn by trial and error.

I hope I have helped you by describing these assemblers. If you want any help with assembler language, the MACE SIG-Assembler is still active; come to a meeting and we will try to help you. You don't need to know assembler.

SO WHAT IS A MODEM, ANYWAY?

by Kirk Revitzer

It occurred to me that a lot of really new users might be wondering what a modem is and why they do or don't need one. So I'll start from scratch.

A modem, which is actually a MOdulator-DEModulator, is a device that connects your computer to the telephone and allows you to 'talk' to other computers around the country. Here are some common expressions and their meanings:

Acoustic modem - You insert the handset of your telephone into rubber cups.

Direct Connect - The modem plugs directly into a modular telephone jack.

Interface - A device to connect the modem to the computer.

Amodem - A terminal program for the Atari computer that supports ATASCII mode and XMODEM transfer.

TELELINK - A relatively useless terminal program by Atari. It does not support file transfer or ATASCII. This is what is built into the Atari 1030 modem.

ASCII - The standard 'character set' computers use to talk to each other.

ATASCII - The Atari-ASCII character set. In this mode Ataris can communicate with each other using inverse video and control graphics. On a BBS this is known as "ATARI MODE"

XMODEM PROTOCOL - An error checking method of transferring files from one computer to another or from a BBS to a computer terminal. XMODEM is a feature included in Amodem and the MPP Smart Terminal.

AMIS BBS - Atari Message and Information System. It's the standard format for just about every Atari Bulletin Board System in the country. MACE operates the original AMIS BBS!

SYSOP - The SYStem OPerator of a BBS.

So there's a little description of what it all means. This is a bare bones beginner's dictionary but it should help get you started. For more information on various types of modems and interfaces see "Modems and Interfaces" in the December '84 issue of the MACE Journal.

So what's in it for you? Well there are all kinds of things to do! First, there are the Atari BBSs. A BBS offers a message base where you can read and leave messages for other users around the country or within your own club. There are hundreds of great public domain files for the taking and the joy of uploading a file to the BBS. Stuck on a problem, need help on an adventure, just leave a message.

There are also a lot of pay services available such as the Source or CompuServe or Dow Jones. But the point is, if you get a modem the "What can I do?" question will almost never get answered. The possibilities are almost endless!

So with that, I think I'll boot up my TELE-CHESS program and call a friend. See ya next month.

AGENDA

February 19th

Officers' reports

Group discussion led by Kirk Revitzer
President of MACE

Activision's Decathlon demo, by Eric Schiffer

Party Quiz demo, by Mike Lechkun

Decision software demo, by Scott Garland

Home Energy Analysis demo

MORE AMODEM!

by Kirk Revitzer

I hope you wanted to know how to send a file from one Atari to another using Amodem because that's what I'm going to talk about. First, the modem itself should be in full duplex. There really isn't any need to adjust that at all. Half or full duplex is an option on the Amodem program and that's what we'll use.

So, now the initial check list.:

1. Boot up Amodem
2. Set the baud rate (300/1200)
3. Set ATASCII mode
4. Set half duplex

Decide which modem will answer and which will originate and establish your connection. At this point you should be able to type to each other. To send or receive, use the S or R option of Amodem just as you would with a BBS. When the receiving end says he's ready watch for him to push his START key. The receiving end MUST push START first. When he does you will see a small white line on your screen. Then, you push START. If you have done all this as you read this article you'll notice your file is now being sent. Gee, that was easy!

Now for some other stuff:
Do you have a 1030 modem and wonder why you have problems with an MPP BBS? Well, it's not your fault. The MPP set for 300 baud is actually running about 305 baud. Unfortunately, this is not within the tolerance of a 1030 modem. If an MPP sets its baud rate to 297 or 298 the connection will work okay.

Here's a little tid-bit hot off the BBS. It seems our Commodore friends have developed a terminal program that will emulate an Atari! So if you Sysops see someone in ATASCII mode you'll just have to wonder who it really is! It seems they can send the ATASCII <RETURN> and read the control & inverse video but I don't think they can send it. Oh well, back to the drawing board!
'Til next month...

CASSETTE CORNER

by Mike Landis

It's the beginning of the year and time for a refresher course in the 410 & 1010 program recorders.

HANDLING CASSETTES:
Be careful with cassettes; they can be easily damaged. Avoid touching the surface of the tape itself. No matter how clean your skin is, natural oils will contaminate the tape. Make sure you put tapes back in their cases when they are not being used. Never store them in hot areas, direct sunlight, or near magnetic fields.

SELECTING BLANK CASSETTES:
The Atari recorder uses only audio cassettes - never digital cassettes. You can't go wrong with the best quality normal-bias tape. Good quality tapes will work too, but avoid cheap bargain cassettes. They tend to jam up after a while, rendering your valuable programs inaccessible.

LABELING CASSETTES:
You should label every cassette with information about the programs it contains. This prevents the headache of searching through cassette after cassette for the program you need.

WRITE-PROTECTING CASSETTES:
Each cassette has two notches in the rear edge. When the notches are uncovered, the recorder can sense the holes and will not record on the cassette. New blank cassettes have tabs covering the holes so the tape can be recorded on. You can protect important programs by knocking out the correct tab and exposing the hole. Later, if you want to record over a protected tape, simply cover the hole with tape.

Each cassette has two sides to it. One notch protects one side, while the other notch protects the other side. To determine which notch is correct, hold the cassette so the exposed tape is toward you and the side you wish to protect is facing up. Remove the tab on the left side to prevent recording over the side facing up.

THE TAPE COUNTER:
The MACE cassette library tapes always start with the menu. Make sure the cassette is rewound, then set the counter to zero (000). If you want to bypass the menu and go right to the first program, start from zero (000) and advance the tape to 30 and then CLOAD.

If you put a new tape into the recorder, don't assume it is rewound. Depress the REWIND lever just to be sure, then reset the counter.

NOTE: Tape counter speed varies from one recorder to the next. Thus, tape counter readings noted on one recorder may not match those of another. It is always recommended that you change the numbers on the menu or tape listings to match your recorder.

I hope I have given you some insight into your wonderful animal. Next month, we get technical! We will talk about LOADing a program, SAVEing a program, One-Step LOAD and RUN, chaining programs and, yes, even program recording formats. Wow! I can't wait!!!

AN APPEAL

by Scott Garland
MACE Program Coordinator

My job as Program Coordinator of MACE is to arrange a meeting for each month. As such, I demo some software, arrange for demos by other people, and have talks on topics of general interest. MACE is a user group, and therefore all of us, as members, must do our part. I do my part by arranging and running these meetings; it is up to you to do your part by coming early and setting up chairs, staying late and putting away chairs, or volunteering to do demos. (By the way, if anybody out there could do a demo of the new SynTrend, SynGraph, SynFile, or SynCalc, please offer your services.) My phone number is 851-9453. Feel free to call me any time between 2:30 pm and 9:00 pm with suggestions, offerings of help, or any questions. I can't do it alone; I need your help, as members of MACE.

WHATS NEXT?

FOR THE INDEPENDENT THINKER

WHEN YOU CHOSE ATARI, YOU CHOSE RIGHT! RIGHT?

- BEST VALUE
- MOST FEATURES FOR THE DOLLAR
- DID JUST ABOUT EVERYTHING YOU WANTED

NOW YOU WANT MORE... MORE FOR THE BUCK... MORE MEMORY... MORE SPEED... PLUS ONE IMPORTANT MUST... COMPATABILITY WITH THE INDUSTRY STANDARD!

SANYO 550 SERIES COMPUTERS

ARE THE SMART CHOICES FOR THE NEXT STEP.

QUALITY & DURABLE... LESS THAN 1% RETURN RATE
SECURITY OF SUPPORT... BACKED BY A $13 BILLION CO.
COMPATABILITY... WITH IBM: HIGH LEVEL
WITH ATARI: WE'LL SHOW YOU HOW TO TRANSFER PROGRAM FILES (TEXT) TO SANYO.

TYPICAL CONFIGURATION:

UNDER $800.00

BUILT-IN 360K DISK DRIVE
128K RAM MEMORY
BUILT-IN PRINTER INTERFACE
MONOCHROME AND RGB COLOR OUTPUT
NUMERIC KEYPAD AND FUNCTION KEYS
WORDSTAR & CALCSTAR

CENTER FOR COMPUTER RESOURCES OF THE FAMILY COMPUTER CENTER 4 BLOCKS E OF GREENFIELD
3895 W. 12 MILE RD · BERKLEY · 543-0520
HOURS: MON- CLOSED TUE-FRI: 10-7 SAT: 10-5 SUN: 1-5

BEAMRIDER

by Activision

Reviewed by Scott Garland and Michael Schiffer

Beamrider by Activision is a Tempest-type shoot-em-up game in which you protect the "Earth-Space" by blasting aliens descending five vertical beams. You are equipped with unlimited "laser-lariats" which destroy only certain enemies; the others may be eliminated by your torpedos, of which you possess three per sector. "Yellow rejuvenators", when caught, will add a ship, but can kill you after being shot. A sector is cleared after killing fifteen "enemy saucers", although countless obstacles, such as "space debris" and "bouncers" will hinder you. At the end of each sector, the mother ship appears, giving you a chance to destroy it for a bonus, which is by no means easy. Each successive sector brings faster action and new (and improved!) enemies.

Although slow at first, the game is devilishly fast-paced and challenging in later sectors. After playing it a number of times with friends, our opinion is this: BEAMRIDER is addictive and exciting. Although not a complex game, BEAMRIDER is worth purchasing.

EXODUS: ULTIMA III

by Lord British
Origin Systems, Inc.

Reviewed by Michael Schiffer

EXODUS is third in a series of critically acclaimed graphic role-playing games by Lord British. It allows up to four characters of varying races (such as elves, dwarves, and bobbits (sic)), and classes (such as clerics, wizards, and thieves). Your quest is not known to you at first, save for the nebulous indication that evil has returned to the world of Sosaria and that your party has been commissioned to find out how to stop it. You must travel about the countryside, fighting monsters with sword and spell, asking questions in towns, and exploring dark dungeons. Each character class is given different powers, and supplied with minimal armor, weapons, gold, and provisions. As each character gains experience, he, she, or it also gains power, but the monsters and puzzles grow more difficult to overcome.

EXODUS includes a player reference card, an instruction book (which still requires you to figure some things out), a book of clerical spells, a book of wizard spells, and a beautiful cloth map of Sosaria. The graphics are rather spartan, but give a good idea of what is going on. It approximates all of the standard features of a game such as Dungeons and Dragons(tm), and you don't need to get a group of people together in order to play. I heartily recommend this for anyone who enjoys role-playing games (and who doesn't?) and has both dedication and stamina.

DECATHLON

by David Crane
Activision

Reviewed by Eric Schiffer

Activision's Decathlon involves all the events from the 100 meter dash to the pole vault to the grueling 1500 meter race. It is hard on joysticks, because while running or approaching you have to move the stick back and forth quickly. This cartridge can be used with any Atari home computer.

The game itself is tiring. It begins with the 100 meter dash which is rather easy. Next is the long jump, then the shotput. The high jump is one of the best; in this event it shows the approach including the angle you need to make the jump. After the high jump is the 400 meter race. Next is another race, 110 meter hurdles, which shows very good graphics. Then comes the discus, the pole vault, and the javelin. Last is the difficult 1500 meter race which seems endless. Many of the events are basically the same, but Activision uses fine graphics and some special highlights including a real-time clock for races and a "strength bar" to tell your relative running or approach time during events. Activision's Decathlon is one of the better sports games on the market.

ATARI & C.E.S.

(from CompuServe)

ANTIC SPECIAL BULLETIN, REPRINTED BY PERMISSION OF ANTIC PUBLISHING INC.

1/2/85
by NAT FRIEDLAND
ANTIC EDITOR

Antic Magazine got a look at the 1985 Atari Computer Line-Up three days before it was due for unveiling at the Consumer Electronics Show in Las Vegas.

This Antic Preview was hosted at the Atari Engineering Center in Sunnyvale by Sig Hartmann, President of Atari's Software Division, and Sam Tramiel, President of the Atari Corp.

Prices for many of the new products were not going to be finalized until 24 hours before CES opens. Manufacturing of the new computers won't begin earlier than March, which means they won't be appearing in the stores until at least May or June.

Atari will be showing three new 8-bit XL-compatible computers at CES. A 128K version, the 130XE, will be priced around $200. The 65XE, the next-generation 800XL, will cost around $100.

The keyboards and cases of these machines will be dramatically changed. They will be smaller versions (no 10-key pad) of the spectacular 16-bit computer housing. Inside the XE computers, the circuitry has been simplified and chip functions have been combined. Sam Tramiel said that this will make the XE a more reliable machine than the XL while maintaining 100% compatibility.

There will also be a luggable 8-bit XE that comes with 128K, a small video screen and a disk drive, the package selling for around $400.

The XE computers will use a new DOS 2.5, which Sam Tramiel said is very similar to the classic DOS 2.0S. The new DOS is necessary so that the XEs can accept the new 500K 3 1/2" disk drives as well as the current 1050 drive. The built-in BASIC has not been changed.

Hartmann & Tramiel brought up and agreed upon the necessity of exchanging DOS 2.5 for the DOS 3 packaged in hundreds of thousands of new 800XLs sold at Xmas. They were very open to Antic's suggestion that DOS 2.5 be uploaded onto the CompuServe Atari SIG and distributed to users' groups as soon as it's finalized.

Hartmann also proposed that Antic assemble users' questions via CompuServe each month and turn them in to him. He promised prompt answers from Atari's top management through Antic. Sam Tramiel was enthusiastic about the arrangement and Antic is starting it right now! Upload your questions for Atari on the Antic Electronic I/O Board and we'll be online with the first round of answers before the end of January.

ANTIC WRITE-INS SAVE PARALLEL BUS!!

Good news that many of us have been waiting for! The XE 128K computer will maintain an open parallel bus for plug-in peripherals. The PBI will even be improved over the current XL format -- with improved timing and a built-in +- 5 volt power amplification.

The last-minute decision to continue the PBI came at an engineering meeting called by Sam Tramiel in response to Antic's write-in campaign on CompuServe's SIG*ATARI.

As for the two new 16-bit machines, they can be described in one word -- phenomenal.

The 130ST is a non-expandable 128K computer and the 520ST carries 512K, that's the only difference between them. They both use the Motorola 68000 chip which is also in the Apple Macintosh. They both use Digital Research's super-fast Macintosh-like GEM user interface, which sits on top of DR's CP/M 68K operating system.

Each model also has 196K built-in ROM containing GEM and CP/M 68K. An Atari mouse will be available to run GEM's icon menus. Both computers will be capable of

running the highly-rated DR version of Logo. C or Pascal will be the development languages of choice.

The state-of-the-art keyboard for the 16-bit machines includes: full Selectric-style key layout, 10-key number pad, cursor pad with Help and Undo keys, and 10 function keys. The entire back of the casing is honeycombed with ports, including: parallel and serial interfaces, MIDI musical interface, PBI, etc.

The one sad note at Antic's preview was that the long-rumored AMIE 16-voice sound chip was not totally ready for production yet. The AMIE, which is said to emulate human speech and singing with unprecedented accuracy, will be incorporated into the '85 Atari line later in the spring and will be identified by adding the letter M to 8-bit and 16-bit models that include it.

Another '85 breakthrough is Atari's sleek, compact 3 1/2" disk drive with 500K storage capacity. Sam Tramiel said this drive will be in the incredibly low price range of $100. Tramiel told Antic that the 500K drive, which uses the same sturdy 3 1/2" disks as the Macintosh, will be compatible with both the 16-bit and 8-bit computers.

"We've had long discussions about the issue of disk compatibility," said Tramiel. "But our conclusion is that we're going to strongly encourage software developers to bring out their new programs primarily on 3 1/2" format. At the low price we'll be selling the new drive for -- frankly 8-bit users would be foolish not to eventually upgrade to a far superior product and format."

However, the current floppy-5 1/4" 1050 disk drive will continue to be manufactured as long as there is a market for it.

In yet another new move, Atari will be showing a wide line of printers and monitors, which will be aggressively marketed for use with IBM and Apple computers as well as Ataris. When asked who was actually manufacturing these new peripherals, Sam Tramiel laughed and said, "Japan Inc."

The Atari printers -- all aggressively low-priced -- will include a 12-cps daisywheel, an 80-cps dot-matrix, and three $50-100 thermal transfer models, including one that prints in color.

The monitors include a 12" monochrome for about $100 -- with a built-in 80-column card for the XE and XL 8-bit series -- and a 14" color monitor for about $200 which will compete with the popular Commodore color unit. A 640x200 resolution RGB Analog model capable of displaying 512 colors will sell for around $300. There will also be a $150 high-resolution 640x400 monochrome model for the ST series.

Atari also plans to show a monitor with a built-in floppy disk drive, plus a low-cost (around $50) 300-baud modem bundled with software for uploading and downloading.

Software president Hartmann described Atari's goal as "switching from a computer game company to a microcomputer company." He said that Atari Software will primarily publish entertaining and easy-to-use productivity programs. "We'll be very selective about games, only publishing games that we believe have a real chance to become major hits."

By the day before CES, Hartmann expected to sign the contract for Atari to bring out an integrated program comparable to Lotus 1-2-3 -- but with code so compacted that it will run on 64K. If this deal is finalized in time, Antic Online will upload a description of the product from CES.

Also under development is AtariWriter Plus, which will include integrated mailmerge and spelling checker. It is to all run on one disk at 128K. Hartmann said he is determined to reach out for user group feedback in the fast-moving software market. He specifically wants major users' groups to act as beta testers for new Atari software.

As a first step towards this, Hartmann said he will bring top Atari executives to meet with users' group officers at the special invitational Worldwide Users Network meeting which Antic will host at our offices during the West Coast Computer Fair in March.

THE SHELL GAME
CRACKING ATARI LOGO

by Ann McBain Ezzell

Last time we discussed some safe locations for storing machine language routines, and a couple of methods to use to put such routines into those locations. Now we can go on to create and store a display list interrupt (DLI) routine that you can use to add interest to your Logo screens.

The procedures which follow are written so that they can be used whether or not you know anything about DLIs. I am planning an article specifically about DLIs for a future Journal, but there isn't enough room for a complete explanation here. Let me just say that a DLI is a way to change certain aspects of the display screen to enable you to (for example) have more colors or different character sets on the screen at the same time.

The particular routine which I have written for this article lets you change the background color part of the way down the screen. You can set the background with the primitive SETBG colornumber, then use these procedures to change some lower part of the screen to a different color. Here is the source code for the machine language routine which does the work:

```
PHA
LDA #198   ;198 = GREEN
STA $D40A  ;WSYNC
STA $D01A  ;COLBK
STA $D018  ;COLPF2
PLA
RTI
```

After saving the value in the accumulator (by pushing it on the stack), it loads the accumulator with the desired color value. It addresses WSYNC to synchronize the color change with the horizontal sweep of the TV electron beam, then puts the new color into COLBK, the background color, and COLPF2, the color of playfield 2 which is also the background color for the Logo text window. (If you leave out this statement, the text window will be a different color, but you will be able to see lines drawn with pen 2 on the bottom part of the screen. Otherwise, they will be the same color as the new background and will only be visible on the top part of the screen.) Next, the accumulator value is restored and the routine returns from the interrupt.

The main procedure to use this DLI is:

```
TO DLI :LINENUM :COLR
CHECK :LINENUM
WRITE.DLI :COLR * 2
PUTROUTINE 13284 :DLI
DISABLE.NMIEN
CHANGE.DLIST 141 :LINENUM
SET.VECTOR
ENABLE.NMIEN
END
```

The two inputs to the procedure DLI let you specify where on the screen the change is to take place and what the new color is to be. The value for :LINENUM must be from 1 to 75 to avoid messing up the display. Values for :COLR are those given on page 26 of the Atari Logo Reference Manual. They can range from 0 to 127.

The first subprocedure checks the value of :LINENUM and halts if you have entered an invalid number:

```
TO CHECK :LINENUM
IF OR :LINENUM < 1 :LINENUM > 75 [PR
[USE VALUES FROM 1 - 75 ONLY] PR [FOR
FIRST INPUT] .CALL 39929]
END
```

(.CALL 39929 returns you to TOPLEVEL.)

The next subprocedure writes the DLI routine, inserting the color value which you have provided as an input to the main procedure, DLI. The input value must be multiplied by two because the Atari uses even numbers from 0 to 254 for its colors, while Logo uses all numbers from 0 to 127.

```
TO WRITE.DLI :COLR
MAKE "DLI (SE [72 169] :COLR [141 10 212
141 26 208 141 24 208 104 64])
END
```

The numbers which make up the list "DLI are simply the decimal equivalents of the assembled version of the source code listed above. A value of 72 is interpreted by the Atari as PusH the Accumulator onto the stack, and so on.

The primitive SE (for SEntence) outputs a list consisting of the elements in its inputs. When there are more than two inputs, you must enclose SE and its inputs in parentheses. Remove the three numbers "141 24 208" if you do not want the text window the same color as the new background. You will then be able to see things drawn with pen 2 on the changed part of the screen.

Next, PUTROUTINE as defined last month is used to store the DLI routine at the beginning of the shape table (location 13284). It will only overwrite shape 0, so you are free to use any of the other shapes.

```
TO PUTROUTINE :LOC :ROUTINE
IF :ROUTINE = [] [STOP]
.DEPOSIT :LOC FIRST :ROUTINE
PUTROUTINE :LOC + 1 BUTFIRST :ROUTINE
END
```

Location 54286 is the non-maskable interrupt enable (NMIEN). Bit seven of this location determines whether or not display list interrupts will be acted upon. Atari Logo has its own "Display List Maker" which automatically sets up the proper interrupts when you switch between Full Screen (FS) and Split Screen (SS). Before you go about setting up your own interrupt, you must turn off NMIEN or risk hanging up the system. This next procedure saves the value in NMIEN, then puts in 0 to disable the interrupts:

```
TO DISABLE.NMIEN
MAKE "NMIEN .EXAMINE 54286
.DEPOSIT 54286 0
END
```

Now you are free to tinker with the display list and reset the vectors which control the interrupt processing:

```
TO CHANGE.DLIST :NUM :LINENUM
.DEPOSIT (14080 + :LINENUM + 5) :NUM
END

TO SET.VECTOR
.DEPOSIT 512 0
.DEPOSIT 513 54
END
```

The offset of 5 in CHANGE.DLIST is there because of the structure of a display list. When this procedure is called in DLI, :NUM is given the value of 141, which is 128 (bit seven set) plus 13 (normal value for graphics 7 display line). This indicates that a DLI should be executed at the chosen screen location. Locations 512 and 513 hold the address (low byte, high byte) of the DLI routine to be executed, which is 13824 (54*256 + 0) in our case.

Once the display list and vector have been changed, you must reset the value of NMIEN:

```
TO ENABLE.NMIEN
.DEPOSIT 54286 128 + :NMIEN
END
```

Adding 128 to the previous value ensures that bit 7, which controls DLIs, will be set. Once this procedure is run, your new DLI will take over and you will be able to use your two color screen however you want.

When you use these procedures, you must call SS or FS first. If you issue either of these commands after running DLI, the Logo Diplay List Maker will take over and the display will change back to its normal state. If you are going to use the DLI routine more than once in your procedures, call SS or FS each time to clear out the old display list.

Remember that lines drawn with pen 2 will not be visible on the bottom part of the screen if you change the text screen background to match the new background. If you need to use pen 2, remove the numbers "141 24 208" from the list in WRITE.DLI.

A DLI could also be used to change one or more of the pen colors at some point on the screen, or even change the turtle color, so that a turtle would change color as it moved up and down the screen. The addresses to use for the various pen and turtle colors are:

```
$D016 (22 208) COLPF0 Pen 0
$D017 (23 208) COLPF1 Pen 1
$D018 (24 208) COLPF2 Pen 2
$D012 (18 208) COLPM0 Turtle 0
$D013 (19 208) COLPM1 Turtle 1
$D014 (20 208) COLPM2 Turtle 2
$D015 (21 208) COLPM3 Turtle 3
```

The numbers in parentheses are the low and high byte equivalents of the hex addresses. Use these in place of the 26 and 208 values in the WRITE.DLI procedure above.

In this case, you will also need to do something about the color of the text window. The procedure as written will make the text window the same as whatever color you put into the altered register. If this is okay, or if you plan only to use Full Screen mode, you can leave WRITE.DLI as it is. The other easy option is to remove the "141 24 208" from WRITE.DLI, which will give you a text window the same color as pen 2. If you don't need to draw with pen 2, you can use the SETPC command to set its color to the background color or whatever color you want to use for the text window.

One other thing to watch when you start changing the color registers and bypassing the Logo Display List Maker is the luminance of the text in the text window. This is controlled by the luminance of pen 1. If you change the color of pen 1 or pen 2 with your DLI and the luminance of pen 1 turns out to be the same as or close to that of the luminance of pen 2, you will not be able to see the characters in the text window. You may have to experiment with different color values to get a legible result.

If you already know how to write DLI routines, you could of course set up multiple DLIs on the screen and format the text window with none of these restrictions. I think that the routines given here will be adequate for most purposes, but if you come up with a fancy display, send it in and we'll publish it.

I hope that you will find these procedures useful in your Logo applications. You should be able to create some interesting effects with the added color capabilities. Next time I will show you some procedures that will let you add text to your Logo graphics screens.

MESSAGE BASE

Anyone knowledgable in R.T.T.Y. for the Atari 800 please call Barb at (313) 231-2531. [There was an article about this in the July '84 Antic magazine. -Ed.]

Last month there was a request for a non-disk translator program for XL computers. Computer Software Services offers the XL "FIX"! which they claim fixes 30% more software than other translators and works with disk, cassette and cartridge software. The price is $49.95 for disk or cassette - ROM cartridge is $69.95. Check out their ads in Antic and Analog.

IT'S OFFICIAL! The first winner of the MACE Journal Best Entry Contest is Jim Wilson for his program Hangman, which appeared in the November 1984 issue. The next contest will cover the February, March and April issues, so get those programs and articles in RIGHT HERE AND NOW.

SIG UPDATE

Here is the current information on active Special Interest Groups. Call the person whose name is listed for more details.

Atarimusic SIG: Mike Lechkun 978-8432

Assembler SIG: Todd Meitzner 542-1752
Meetings are the 1st Thursday of each month. Next meeting: February 7th.

East Side SIG: Mike Simpson 751-7290
Meetings are the 1st Tuesday of each month. Next meeting: February 5th. The meeting will be at 7:00 pm at the Italian Cultural Community Center, 28111 Imperial in Warren (between Hoover and Schoenherr).

FORTH SIG: Tom Chrapkiewicz 562-8506
This group is not just for Atari owners; any FORTH user is welcome.

Graphics SIG: Dan Rubyan 838-5679
Call Dan if you are interested in joining this new SIG.

BIG BIRD'S FUNHOUSE

A Review
by Ann McBain Ezzell

With so much "swillware" passing itself off as educational and entertaining, it is refreshing to find some preschool software which actually is. CBS Software and the Children's Computer Workshop have produced a program which will not only keep your child at the keyboard, but also let her practice memory and sequencing skills. Big Bird's Funhouse is one of several Sesame Street programs aimed at the 3 to 6 crowd, and is well worth the investment.

Big Bird's Funhouse is available in cartridge, disk and cassette to run on all Atari computers, although the membrane keyboard on a 400 might be awkward to use with the EasyKey overlay. I tested the cartridge version, and was pleased that it slipped easily into both my 800 and 800XL, unlike some other third party cartridges which tend to stick. The EasyKey overlay is a piece of flexible molded vinyl which fits over the keyboard and is essential to the play of the games; it shows you where to press. It has pictures of eight Muppet friends plus areas for different commands such as "HIDE", "PLAY" and "LEVEL". The molding and printing were not exactly aligned on the one that I tested, but that didn't affect its use. The molding is made to fit the XL style keyboard, but fit adequately on my 800, although I found on both machines that I needed a couple pieces of masking tape to keep it in place.

Documentation for this program consists of a colorful, 24-page Game Play and Activity Manual and a quick reference How To Play card. About half of the manual is directions for the various levels; the rest is a story about the characters in the game plus suggested activities to reinforce the skills taught by the program. The manual is designed to encourage active participation of the adult with the child. Too often, preschool programs are chosen for their "baby-sitting" capabilities; it's nice to see one which emphasizes interaction.

The graphics and sound in Big Bird's Funhouse are excellent. Each Muppet character is easily recognizable and nicely animated. A tune accompanies each character and helps you to remember them. The five levels of play all involve remembering which Muppet friends are playing Hide and Seek in the funhouse.

In Level 1, Who's Hiding?, you invite three characters who then pop up in the windows. When you press the HIDE key, you see them scampering around the funhouse, blowing trumpets, sliding into a tub of water, and generally having a good time. Then they disappear and shutters on all but three of the windows slam shut. Two of the friends appear; you must remember who is still hiding and press the appropriate key. If you make a mistake, Big Bird comes out and shakes his head "No". After a second wrong guess, the tune for the missing character plays as a hint. With a third incorrect answer, the character pops up briefly in his window. After you have found the missing Muppet, you press the HIDE key for the second round. This time two of the characters are missing. In the third round, all three are hiding. Once you have successfully completed three rounds, Big Bird rewards you with a rollerskating demonstration.

The other levels are variations of Level 1. You can invite more characters to play in Levels 2 and 3; in Level 3 all the windows remain open, so you can't count the remaining windows to help you. In Levels 4 and 5, Big Bird does the inviting when you press the POP-UP key. Both of these levels require that you remember the order in which the friends appear. Level 5 continues adding a friend each time until you fail to match the sequence.

The games progress at a reasonable speed: fast enough to be interesting, but not frantic. Wrong answers are corrected gently and right answers are rewarded with music and animation. Control of the program is simple. You can press the LEVEL key to return to the menu screen, or use the PAUSE key to stop and restart a game in progress. Once a child is familiar with the games, she can (and probably will) continue to play by herself for quite a while. If you have a young child and an Atari, you should have Big Bird's Funhouse.

SECTOR ONE

INTERNATIONAL

ATARI

FREE CATALOG

COMPUTER MAIL ORDER *WORLDWIDE*

37220 TRICIA DRIVE, STERLING HEIGHTS, MI. 48077

ORDER INFO. Phone: 313-978-2208 9am - 7pm. EST

SECTOR ONE *signs of VALUE*

600XL COMPUTER NOW $ 69.95	800XL COMPUTER JUST $118.72	850 INTERFACE LOW $129.20	1010 RECORDER PRICE $ 44.95
1020 PRINT/PLOT. NOW ONLY $ 69.70	1025 PRINTER JUST $199.97	1027 PRINTER LOW $259.92	♥ LEGEND 880 ♥ PRICE $259.95
1030 MODEM ONLY $ 59.76	1050 DISK DRIVE JUST $189.72	TOUCH TABLET LOW $ 45.00	835/1030 ETMODEM DSK SOFTWARE $15
CX-85 KEYPAD ONLY $ 29.80	HAYES S.M. 300 JUST $199.95	HAYES CRONOGRAPH LOW $199.95	HAYES S.M. 1200A PRICE $489.95
♥ MPP-1000 E ♥ ONLY $119.90	MPP-1150 INTERF. JUST $ 79.80	MPP-1152 BUFFER LOW $ 14.96	MPP- MICRO PRINT PRICE $ 63.84
SAFT 200 WATT EP ONLY $437.57	SAFT 400 WATT EP JUST $597.17	COLOR DISKS SSDD 12 RAINBOW $20	10 DISK MAILERS PRICE $ 6.84
1020 PRINT. PENS ONLY $ 1.97	1025 PR RIBBON JUST $ 2.22	1027 INK ROLLER LOW $ 3.91	LEGEND PR RIBBON PRICE $ 9.95

DON'T SEE IT? PLEASE CALL - 313-978-2208

BANK-SWITCHING
with the Atari Computer

by Kumar Bhatia
Essex Regional Atari Computer Enthusiasts

Let's say you are a software author and have just finished the next blockbuster game of the century! You intend to sell it in cartridge form but are afraid that someone will be able to copy your masterpiece. You have two options. One is never to release the program (don't pick this one!!); the other is to plant one heck of a protection scheme around your software. There are basically two ways to protect cartridge software: SOFTWARE and HARDWARE. If you choose just the software route, like the majority of software houses do, then you should write and encrypt a subroutine which constantly writes to the cartridge locations. If someone were to transfer the cart to a file and upload it into RAM, the program would literally self-destruct when executed. The drawback is that this ROM process can be emulated, so software alone is a poor choice of protection. With hardware it is possible to devise schemes which would baffle even a top notch bandit. A well known company has done exactly this; they call their products "SUPERCARTRIDGES". With special hardware, these cartridges can "bank-switch" 16K of memory in an 8K space.

These can be considered to be the "second generation" of cartridges, and although the principle is new to Atari computers, it has been used for a while in the Atari VCS game machine. "Bank-switching" is nothing more than the concept of overlaying a region of memory with more memory by toggling between different banks. The rest of this article describes in detail how you can transfer your own piece of software into a bank-switching cartridge. It is assumed that the finished program is 16K in length and is to be converted into an 8K memory efficient cartridge.

The first step is to organize your software into specific banks. A proposed layout is given in the theory section. The entire 16K program is to be broken up into four banks, of which Bank #1 is constant and is called the CONTROL bank. The other three banks, each 4K in size, are to share the memory space $A000-->$AFFF (one at a time, of course!). The control bank should be responsible for placing the bank needed into memory region $A000-$AFFF for operation at power-up. The program will be stored in two 2764 EPROMS (8K x 8).

BASIC OBJECTIVE: In essence, you want to have 16K of data sitting in an 8K region. The proposed design's memory map is as follows:

$B000-->$BFFF Bank #1 (Constant)
$A000-->$AFFF Bank #2 ($D500)
$A000-->$AFFF Bank #3 ($D503)
$A000-->$AFFF Bank #4 ($D504)

As you can see from the memory map, there is a constant 4K bank sitting at $B000-->$BFFF, while three more banks of 4K each share the area $A000-->$AFFF. When you power-up with the cicuit shown in Figure 1, the cart defaults by putting Bank #2 at $A000-->$AFFF and Bank #1 at $B000-->$BFFF. For those with wild and creative minds, the hardware supports the ability to bank away Bank #1. Beside each bank number you will find the appropriate address with which it is possible to toggle that piece of EPROM in or out. The circuit is wired with the R/W (READ/WRITE) line of the memory so that when you perform a WRITE command to location $D500, Bank #2 will immediately occupy the 4K region starting at $A000. Similarly, if you were to do the same with locations $D503 and $D504, then Bank #3 and Bank #4 respectively would toggle in that memory space. (Note: For practical purposes, I chose to get the R/W line from the OS board. However, this signal is available from the cartridge slots if you wish to construct a PC-board for such a circuit. Consult the Atari Technical Manual for further details.)

Now, let's translate the above into hardware. First of all, you'll have to burn your own set of EPROMS with the sofwtare. You'll need to burn two 2764 8K EPROMs. You can accomplish this with an average quality eprom burner without great difficulty. A theoretical circuit is shown in Figure 1 and can be built wire-wire or on a PC-board. To test the

circuit, plug in the cartridge and power up. If you have written the driving software correctly, the banks should switch smoothly!

BANK-SWITCHING CARTRIDGE
PART LIST:

2 x 2764 EPROMS (450 ns or faster)
1 x CD4514 Multiplexer/Latch IC
1 x 74LS02 NOR Gate IC
1 x 74LS08 AND Gate IC
1 x 10K 1/4 watt resistor
1 x PC BOARD for housing EPROMS*
1 x Small piece of perforated board (.01 inch spacing)

* Note: The PC BOARD for holding the two 2764 Eproms is available for $7.50 plus postage from:

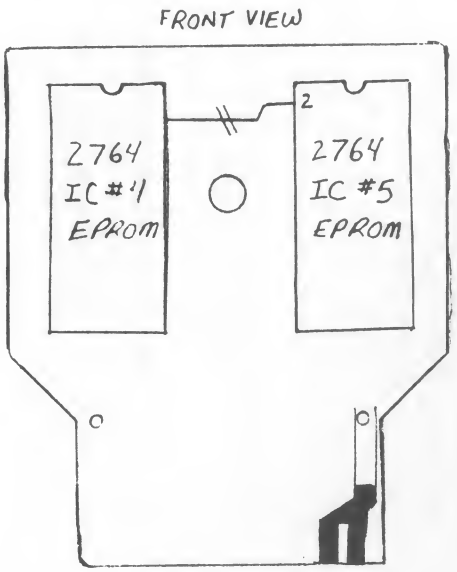
CONVOLOGIC Inc.
421 Bay Tree Lane
Longwood, FL 32779
(305) 869-6630

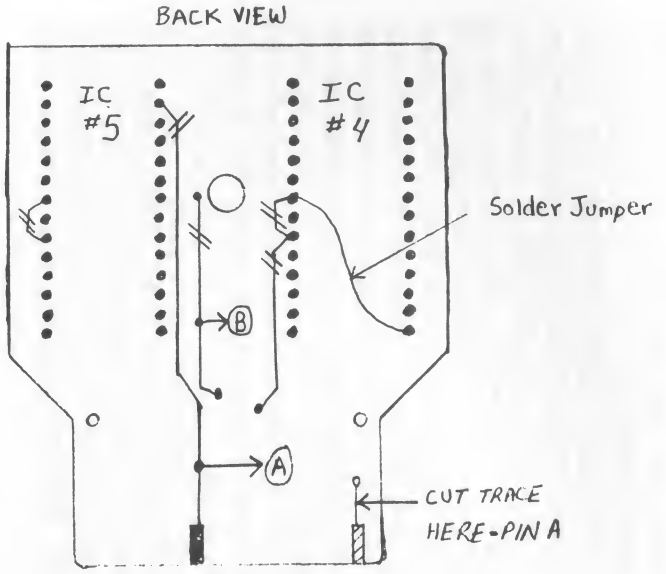
With a sharp X-ACTO knife, cut the trace going to pin 2 on each EPROM socket as shown. Cut the traces going to pin 20 on both sockets and solder to these connections as shown in Figure 2. Finally, cut the trace going to "A" (on the solder side of the board-labelled) on the PC board connector. This will allow you to disable only the top 8K of RAM when this cartridge is inserted. The actual circuit can be built on perforated board due to its simplicity. One last connection must be made to access the $D5xx select line. Although this line is available through the cartridge slot, the above PC board does not provide a trace for a connection. Therefore, the signal must be "stolen" from the 74LS138 IC on the Atari OS board (first slot in Atari 800). This select line is available on pin #10 on IC Z401. If you have an Atari 400 you will have to open the computer case to gain access to solder to pin #10 on IC Z105 (74LS138). To run the cart, insert the finished product into the left slot with the chips facing AWAY from the computer and power up.

THEORY OF OPERATION: The theory of operation for the schematic in Figure 1 is quite straightforward. The circuit consists of two 8K EPROMS (Electrically Programmable Read Only Memory devices), thereby providing the necessary 16K data storage for the object code. With the aid of two more integrated circuits (IC) we can perform bank-switching. IC #1 is a CD4514, 4-BIT LATCHED/4-to-16 Line Decoder and its job is to act as an address decoder for the unique addresses $D500, $D503 and $D504. When any address in the $D500 page ($D500-->$D5FF) is selected the strobe on the latch (pin 1) goes LOW thereby latching the data (actually address lines) fed into that memory page. For example, if you access $D500, either with a READ or WRITE command, IC #1 will force pin 11 to go HIGH. This activates IC #4 to select Bank #1. If you access $D503, then IC #4 will come back into use again BUT address line A12 on IC #4 is forced to be HIGH, selecting Bank #2. IC #2 and IC #3 make sure that no bus conflicts occur while you are bank-switching. Note that whenever you access the region $B000-->$BFFF, Bank #1 will always toggle back into its place.

I hope that you have gained some new knowledge on this topic and utilize it in your next project. HAPPY HACKING!

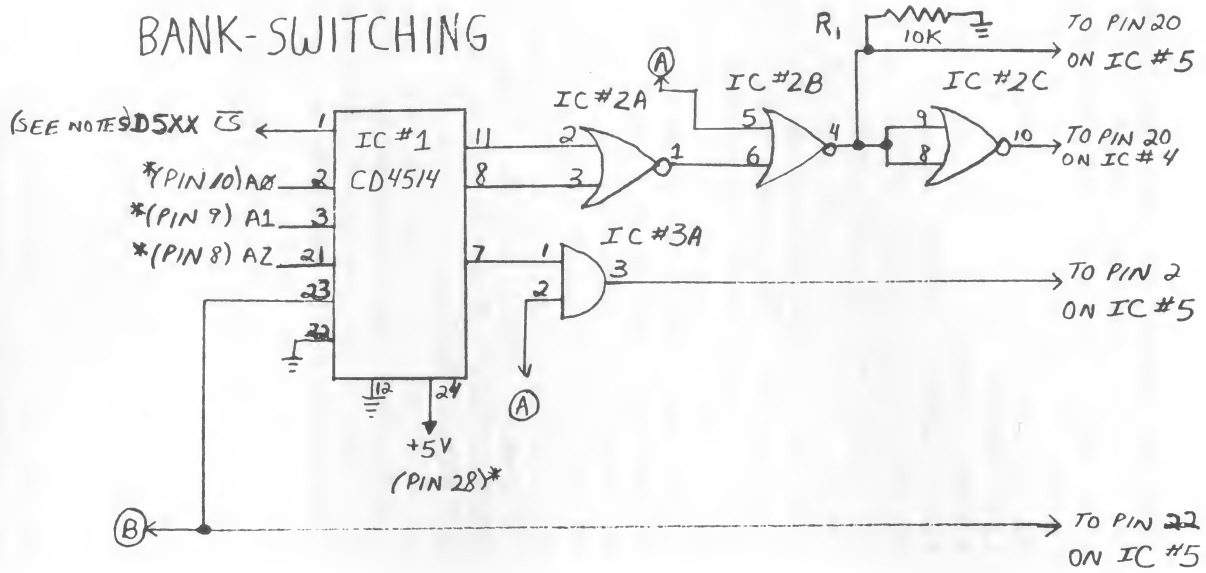
FIG. 2





\\ - DENOTES CUT TRACE
* - CONNECT TO ANY EPROM SOCKET

BANK-SWITCHING



PARTS LIST

IC1 - CD4514 4-BIT TO 16 WITH LATCHED OUTPUTS
IC2 - 74LS02 QUAD 2-INPUT NOR GATES
IC3 - 74LS08 QUAD 2-INPUT AND GATES
IC4, IC5 - 2764 (8KX8 EPROMS)
R1 - 10K 1/4 WATT RESISTOR

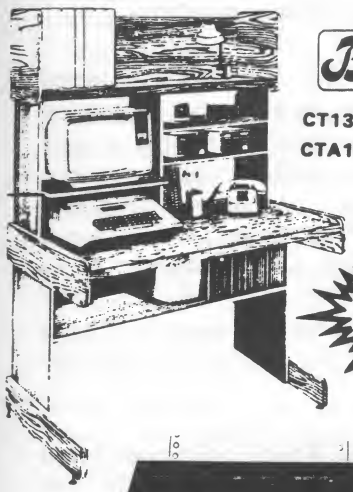
POWER PINS

	+5	GND
IC1	24	12
IC2	14	7
IC3	14	7

FIG. 1

K.B.

RITE WAY COMPUTER CENTERS



Bush

CT130 *Desk*
CTA131 *Hutch*

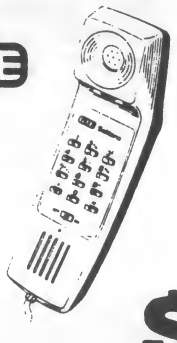
$179

PRINTERS
DOT MATRIX

LEGEND 880 $229.00
LEGEND 1080 $269.00
GEMINI 10 X $269.00
LETTER QUALITY
AXIOM ELITE 5 $299.00
POWERTYPE $399.00

COMPUTER
PAPER
$19.95 * CASE
15or20 lb. PAPER

GTE
TELEPHONES
$5.95

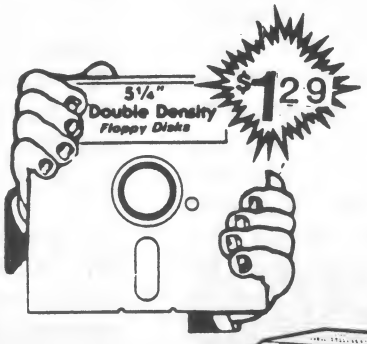


$69.95
Basic Low-Back Chair

VISA MasterCard

Software Always Discounted

20% OFF LIST



SOFTWARE & HARDWARE FOR
IBM
apple ATARI
commodore

150ns 64K
RAM CHIPS
$3.95

SOFTWARE CLUB
$19.95 PER YEAR

RENTALS **$1.99** AND UP

RITE WAY
CASH REGISTERS •COMPUTERS

751-2454
8242 12 MILE RD.
WARREN, MI. 48093

562-3178
22027 MICHIGAN AVE.
DEARBORN, MI. 48124

483-5410
1514 WASHTENAW
YPSILANTI, MI. 48197

MON—FRI
10-7
SATURDAY
10-5

We welcome Visa and MasterCard.

PAGE ZERO

by Ann McBain Ezzell

MACE members come in many varieties, from absolute novices ("How do you get this thing out of the box?") to veteran hackers ("Connect the frim-fram to thromdibulator, and away you go!"). We obviously can't meet everyone's needs, but here is the latest attempt: a column for beginners. I will assume that you have figured out how to open the carton and get all the assorted parts plugged in and humming, and are now ready to tackle some programming. This month, I will talk a bit about computers and numbers so that you won't look totally bewildered when your friends start rattling on about bits and bytes and hexadecimal notation.

First, let's think about the sort of numbers most of us use all the time: 0, 1, 2, 3, and so on. We have ten digits (0 through 9), and can count up to nine objects using just one digit. Once we hit ten, however, we switch over to two digits: 10. This means (reach back into your grade school memories) zero ones and one 10. Using two digits, we can count up to 99 objects, then we need to add another digit for one hundred or more: 100. This means one 100, zero 10s and zero ones. 100 is equal to 10 x 10, which can also be written 10^2 (ten to the second power, or ten squared). 1000 is 10 x 10 x 10, or ten to the third power (10^3). You can see that each "place column" represents a power of ten. We call this number system "base ten".

Base ten is fine to use if you have lots of fingers handy for keeping track of things, but what about a computer? It deals in electrical voltages, and its basic components can only handle two states: high and low (which we will call 1 and 0). So how many numbers can we express using just two digits? It's obvious, isn't it? Just two: zero and one. If we want to write the number two using this binary notation, we will have to use "10", which is one 2 and zero ones. Using two binary digits, we can express four different numbers: 00 (zero), 01 (one), 10 (two) and 11 (three). So to get to the number four, we would have to add another digit: 100. Four is equal to 2 x 2, or 2^2. The next digit added would be 2^3 or 2 x 2 x 2, which is eight. So the number eight in binary notation would be written: 1000.

Now, a brief detour to the nomenclature department: My trusty dictionary defines a bit this way: "In information theory, a unit equal to the information content inherent in a choice between two equally probable alternative messages or symbols, as between a 'yes' and a 'no' or a dot and a dash". It is derived from the words "b(inary) (dig)it". For our purposes, think of a bit as being either a one or a zero: one of the digits in a binary number.

You may have heard the Atari referred to as an "eight-bit machine". This means that the Central Processing Unit (CPU) can handle information in eight bit chunks (these chunks are called bytes). How large a number can we express using eight bits? One digit will give us two numbers (zero and one), which is 2^1. Two digits gives us four numbers, or 2^2; three digits give us eight (2^3). With eight digits, then, we should be able to express 2^8 different numbers. Two to the eighth power equals 256, but remember that we start with zero, so a byte can be anything from 0 to 255.

To express numbers larger than 255, you have to use more than one byte. If you have done much reading about your Atari, you may seen references to "high" and "low" bytes, or Most Significant Byte (MSB) and Least Significant Byte (LSB). When two bytes are used to express a number, one represents the "ones column" and the other represents the "256s column". This latter byte is the "high" or Most Significant Byte. To convert such a number to decimal (base 10) notation, you would take the value of the high byte and multiply by 256, then add the value of the low byte.

High byte = 54
Low byte = 6
Decimal equivalent = 256*54 + 6 = 13830

To convert a decimal number into its high and low bytes, divide the number by 256 and take the integer (whole number) part, which is the high byte. The low byte is the number minus 256 times the high byte:

53674 / 256 = 209.664
High byte = 209
53674 - 256*209 = 170
Low byte = 170

The Atari usually stores numbers greater than 256 in two consecutive addresses, with the low byte stored first. You will often see something like this in programs:

```
100 SCREEN = PEEK(88) + 256 * PEEK(89)
```

Locations 88 and 89 together hold the address of the screen display memory. This program line fetches the low and high bytes and combines them to be stored in the variable "SCREEN".

Binary notation may be fine for the computer, but it's rather cumbersome to use. Who wants to write "01100110" for the decimal number 102? Also, converting between decimal and binary can be awkward. There is another form of notation which is often used with computers: hexadecimal. The etymologists among you will immediately realize that this system uses 16 as a base, just as decimal uses 10 and binary, 2. But wait! What are we going to use for digits if we have to express numbers up to 15 using only one digit? Hex notation uses the letters A through E for decimal 10 through 15.

Using only two hex digits, we can express any number up to 255 (hex numbers are often prefixed with a "$" to differentiate them from decimal numbers):

0	$00
10	$0A
16	$10
92	$5C
255	$FF

To convert between hex and decimal, just remember that the rightmost column in a hex number is the ones column, and the next column is the 16s column. A four digit hex number also has a 256s column (16^2) and a 4096s column (16^3). Four digit hex numbers can be broken into high and low bytes (the high byte being the two lefthand digits) for easy conversion to decimal notation.

One more new word, and I will leave you to digest all of this until next month. The word is nybble, which is (what else?) half of a byte. Think of the hex notation for a byte: it has two digits, each of which is a nybble (high and low, left and right). Thinking of bytes in terms of nybbles makes conversion between binary and hex a cinch; divide the eight bits into two sets of four and convert each into a number from 0 to E (decimal 0 to 15).

For example, take the binary number 01011100. Divide it into two sets of four: 0101 and 1100. The first set is equivalent to 5 (zero 8s, one 4, zero 2s, and one 1); the second equals 12, or C. Therefore, this number can be written in hex notation as $5C.

So there you have it: bits, bytes, and nybbles; decimal, binary and hexadecimal notation. I hope that this will help you in your Atari explorations, and I'll be back next month. Any suggestions for future columns should be sent to me at the MACE PO Box.

SAVE SOME MONEY

by John J. Marhevko

The day that my Atari 810 disk drive broke down I thought OH NO! I bet it will cost at least $25.00 just for someone to look at it, plus some more money if some part went bad. And then I'm without my disk drive for probably a week or two. GADZOOKS!!! No playing Pac Man for me or word processing for the kids doing their school work!

With a little common sense and thinking I tried another power pack. Sure enough, the only thing that went bad was the power pack. But now I would have to buy a new one for $20.00! Here's thanks to the nice people at Rite-Way who told me to open the bottom of the power pack with a screwdriver and check the fuse. Sure enough, the fuse was burned out. It took me only ten minutes to go to the store and buy a 35 cent fuse and put the whole thing back together. Now the world is safe again because I'm back to killing all those nasty creatures on the TV.

ALL ABOARD!

Sometimes all the functions on a bulletin board can be confusing, especially if it's your first trip across the wires. You are faced with a long list of letters, with no idea of what each does. (No, "A" does not stand for "AAAARRRRRGGGGHHH!!!") Most AMIS format boards use more or less the same functions, so here is the help file from MACE West (313-582-0657) to get you started.

A - ASCII/ATASCII switch. Changes control character sets between ASCII and ATASCII. To use, request 'A', set your terminal program to ATASCII and when you are returned to the BBS, press <RETURN>.
B - Bulletin reprints the bulletin sent at log-on to MACE WEST.
C - Configuration of system. Describes major pieces of equipment currently used to run this board.
D - Download file. Send a program to you. Use this function in ATASCII mode, or inverse text will not be sent correctly. If unsure of XMODEM protocol, respond 'N'. All BASIC files are in LISTed format.
E - Enter a message into MACE WEST. Pressing 'Q' after entering a message will abort the message. Other 'E' subfunctions are self-explanatory.
F - List files that may be sent with the 'D' (Download) function. Files are in double density so they are actually twice as large as they appear.
G - Goodbye. Exit from MACE WEST.
H - Help. Prints this list.
I - Index of download files. A very short description of programs available in the Download files.
K - Kill a message. Use this to delete a message from the file. A password may be necessary if one was used at message entry. The sysop will kill a message with a forgotten password if a message is left at log-off.
L - Line feed on/off. Normally off. For terminals that need an extra line feed to advance the line.
M - Message base. Use commands E, Q, R, and S within this subsystem. Using 'M' within the message base will return you to the main menu.
N - News. Contains club news and interesting articles from outside sources.
O - Other Atari BBSs listed by area code.
Q - Quick scan. An abbreviated scan. Prints only the subject of the messages.
R - Retrieve messages. Allows you to read messages contained in the message base.
S - Short scan. Prints the name of message sender, the intended recipient, subject heading and date of the message.
T - Time and date. Gives you the current time and date.
U - Upload. Use to upload a file to MACE WEST from your terminal. If unsure of XMODEM protocol, answer 'N' to the prompt.
W - Welcome. Reprints the welcome message sent at log-on.
X - Expert user. Shortens some system prompts.
Y - Yell for system operator for chat.
? - Short list of BBS commands.

CTRL<S> Pauses the display.
CTRL<Q> Continues the display.
CTRL<N> Goes on to the next message.
CTRL<X> Aborts current send.

ODDS AND ENDS

Paul Wheeler reports that he has had calls asking about the possibility of forming a morning or afternoon SIG group for MACE members who work the second and/or third shift and can't make it to the regular evening meetings. If you are interested in such a group, send your name, address and phone number to the Journal Editor at the P.O. Box, or leave the information in the Suggestion Box at a meeting.

COMING SOON!

MACE BBS will be revving up to 1200 bps in the near future. No more dreary slo-mo downloads for you speed demons.

Effective March 19th, certificates for free program disks/tapes will only be honored for 60 days from the date of issue. If you have some old ones hanging around, trade them in now. Also please remember to bring your MACE membership card when you purchase from the libraries; they are open to MEMBERS ONLY.

XMODEM FILE TRANSFER PROTOCOL

By Larry Jordan

[This article was derived from material contained in a book written by Larry Jordan and Bruce Churchill to be published Summer '84 by The Brady Company. This article was downloaded from the G.T.I.A. BBS in Kitchener, Ontario, Canada. Our thanks go to Steve Gauthier and the gang in the great white north! -Ed.]

When transferring files between computers using the telephone system, there is always the chance that electrical noise will result in data transmission errors. To ensure proper transfer of files it is necessary to detect data transmission errors and to retransmit data that contains errors. Most people think that asynchronous parity error detection provides that capability. It does not. Parity error detection does tell you when a data transfer error has occurred, but it is up to you to retransmit the data to correct errors. The problem is that parity error detection is not actually performed by most ATARI communication packages. If a package does perform the error detection, it may not inform you of errors in such a way that you know to immediately retransmit the data. To ensure "error-free" data transfer you need a protocol file transfer technique.

A protocol is a set of rules and conventions that apply to a specific area of communications that allow participants to properly communicate regardless of the hardware brand or software package being used. The protocol file transfer is a set of rules for transferring files which specifies a set of ASCII handshaking characters and the sequence of handshaking required to perform certain file transfer functions. Protocol handshaking signals allow communication software to transfer text, data and machine code files, and to perform sophisticated error-checking. The handicap in using protocol file transfer techniques is that the computers on both ends of the communications link must be using compatible software; there is no standard that controls these protocols and almost all communication packages that have a protocol file transfer option use a protocol unique to that package. This means that a business or group of people must standardize its microcomputer communications software to take advantage of protocol transfers.

The Ward Christensen XMODEM protocol is one specific file transfer protocol that may become a default standard in personal communications because of its widespread use on bulletin boards and because of its inclusion in low cost personal computer communication packages such as AMODEM 4.2. It has not gained widespread acceptance in business communication packages partly because the protocol is public domain; most business communication package designers use unique protocols to force businesses to use their software on both ends of communication links. By providing you with this insight into protocol transfer and explaining in detail the operation of the XMODEM protocol, I hope to add momentum to the development of a "standard protocol" whether it be the XMODEM model or some other model. Users of communication software deserve a standard protocol that will allow them to use the technique with any microcomputer regardles of the software packages employed.

The XMODEM protocol is illustrated in Figure 1. As you can see from that figure, XMODEM does not begin the transfer of data until the receiving computer signals the transmitting computer that it is ready to receive data. The Negative Acknowledge (NAK) character is used for this signal and is sent to the transmitting computer every 10 seconds until the file transfer begins. If the file transfer does not begin after 9 NAK's are sent, the process has to be manually restarted.

After a NAK is received, the transmitting computer uses a Start of Header (SOH) character and two block numbers (a true block number followed by a 1's complement of the number) to signal the start of a 128-byte block of data to be transferred then sends the block followed by an error-checking checksum. The checksum is calculated by adding the ASCII values of each character in the 128 character block; the sum is then divided by 255 and the remainder is retained as the checksum. After each block of data is transferred, the

receiving computer computes its own checksum and compares the result to the checksum received from the transmitting computer. If the two values are the same, the receiving computer sends an Acknowledge (ACK) character to tell the receiver to send the next sequential block. If the two values are not the same, the receiving computer sends the transmitter an NAK to request a retransmission of the last block. This retransmission process is repeated until the block of data is properly received or until 9 attempts have been made to transmit the block. If the communications link is noisy, resulting in improper block transmission after 9 attempts, the file transfer is aborted.

XMODEM uses two block numbers at the start of each block to be sure the same block is not transmitted twice because of a handshake character loss during the transfer. The receiving computer checks the transmitted block to be sure that it is the one requested and blocks that are retransmitted by mistake are thrown away. When all data has been successfully transmitted, the transmitting computer sends the receiver an End of Transmission (EOT) character to indicate the end of file.

The XMODEM protocol offers the Atari several advantages over other protocols and file transfer methods. First, the protocol is in the public domain which makes it readily available for software designers to incorporate into a communications package. Second, the protocol is easy to implement using high level languages such as BASIC or Pascal. Third, the protocol only requires a 256-byte communication receive buffer which makes it attractive for Atari owners. Fourth, the protocol allows a user to transfer non-ASCII 8-bit data files (i.e., OBJ, EXE and tokenized BASIC) between microcomputers because it calculates the end of a file based on file size and uses handshake signals to indicate the end of a file instead relying on an end of file marker character (control-Z) to terminate a file transfer. Fifth, XMODEM error-checking is superior to normal asynchronous parity error checking. The parity method of error-checking is 95% effective if the software on the receiving end checks for parity errors. XMODEM error-checking is 99.6% effective, and the software on the receiving end must check for errors. Parity errors detected also do not result in automatic retransmission of the bad data; XMODEM detected errors result in data retransmission until no errors are detected or until 9 retransmissions have been attempted. Finally, the protocol is used by many bulletin boards and having the protocol in a communications package allows the Atari user to receive error-checked files from these boards.

XMODEM Protocol File Transfer

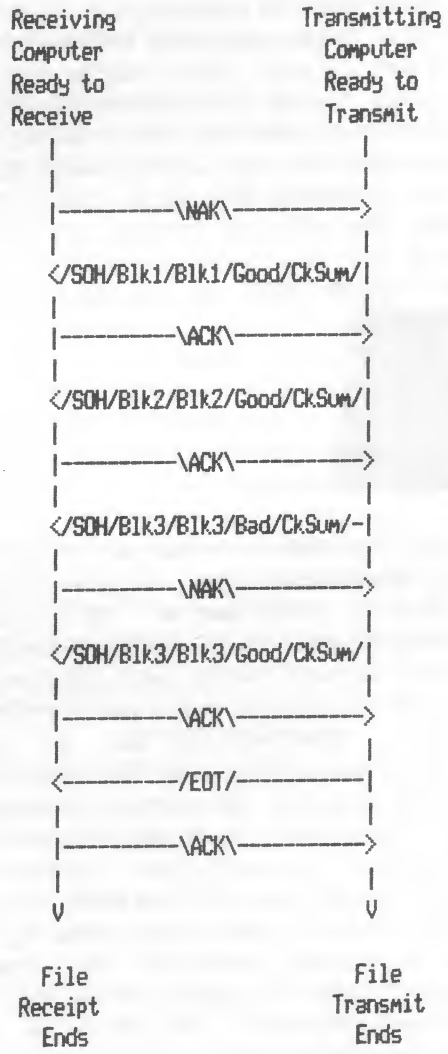


Figure 1

M. A. C. E.

MICHIGAN ATARI COMPUTER ENTHUSIASTS

P.O. Box 2785, Southfield, MI 48037

BULLETIN BOARDS: MACE 978-1685/MACE WEST 582-0657

MACE HOTLINE (Information - Voice Line) 882-7104

PRESIDENT
Kirk Revitzer
Detroit
882-9109
Compuserve: 70346,1642
BBS: 882-5909

VICE-PRESIDENT
Alva Thomas
Detroit
535-3748
Compuserve: 74065,334
BBS: 538-0197

TREASURER
Burt Gregory
Southfield
552-0273

PROGRAM COORDINATOR
Scott Garland
West Bloomfield
851-9453

CORRESPONDING SECRETARY
Sharie Middlebrook
Dearborn
581-5560
BBS: 582-0657

RECORDING SECRETARY
Dino Roggero
Redford
BBS: 531-1701

DISK LIBRARIAN
Dave Zappa
St. Clair Shores
773-8551
BBS: 771-4126

CASSETTE LIBRARIAN
Mike Landis
Clawson
589-1789
Compuserve: 72675,1023

MEMBERSHIP CHAIRMAN
Paul Wheeler
Detroit
538-3649

M.A.C.E. JOURNAL EDITOR
Ann McBain Ezzell
Birmingham
646-4455

NEXT MEETING: 2/19/85 7:00 PM

Southfield Pavilion--Ten & a Half Mile Road and Evergreen

M.A.C.E.
P.O Box 2785
Southfield, MI 48037

$20.00* FOR A ONE YEAR MEMBERSHIP
(*Payable to M.A.C.E.)

M.A.C.E. MEMBERSHIP APPLICATION

Name _____ Phone _____
Street _____
City _____ State _____ Zip Code _____
Company (if applicable) _____
System Description _____ Disk/Tape _____
Suggestions _____
I can help with . . . _____

( )New ( )Renewal
If a Renewal:
MACE #_____

Date _____ Coupon _____
Amount _____ Membership Card Number _____
☐ Cash Expires _____
☐ Check Number

MICHIGAN ATARI COMPUTER ENTHUSIASTS
P.O. BOX 2785
SOUTHFIELD, MICHIGAN 48037